Background Sprite Generation using MPEG Motion Vectors

R. Venkatesh Babu
LSML, Department of EE
Indian Institute of Science, Bangalore.
rvbabu@ee.iisc.ernet.in

K. R. Ramakrishnan
LSML, Department of EE
Indian Institute of Science, Bangalore.
krr@ee.iisc.ernet.in

Abstract

Sprite coding, accepted by the emerging MPEG-4 standard is a very efficient method for representing the background video object. Still this sprite generation is an open issue due to the foreground objects which obstructs the accuracy of camera motion estimation and blurs the generated sprite. In this paper we propose a method for constructing the background sprite with partial decoding of the MPEG stream. Initially the Independently Moving Objects (IMO) are segmented out from the background by clustering the pre-processed motion vectors of MPEG video. The camera motion parameters are obtained from the motion information corresponding to the background region which is used for frame alignment.

Keywords: Compressed Domain, MPEG-4, Sprite generation, Video Mosaics.

1 Introduction

Sprite coding is supported by the new video coding standard MPEG-4 Ver.1 main profile. It provides content-based functionality and low bit-rate video compression. A sprite is a video object that is usually larger than the displayed video and is persistent over time. It is used to represent large, more or less static areas such as backgrounds. Sprites are encoded using macroblocks. The sprite is essentially a static image that could be transmitted only once at the beginning of the transmission. Sprites have been included in MPEG-4 mainly because they provide high compression efficiency in such cases. For any given instant of time, the background VOP can be extracted by warping/cropping this sprite appropriately. Sprite-based coding is very suitable for synthetic objects, but can also be used for objects in natural scenes that undergo rigid motion. Static sprite is generated off line before the encoding process. The decoder receives each static sprite before the rest of the video segment. The static sprites are encoded in such a way that the reconstructed VOPs can be generated easily, by warping the quantized sprite with the appropriate parameters.

Video object segmentation is another difficult task among the research community. There are many semiautomatic and automatic techniques proposed during the last few years with the development of MPEG-4 standard. For most of the sprite coding applications, it is not necessary to segment the foreground object precisely. It is enough to have a coarse foreground object mask for sprite generation.

In this paper we propose a method which can generate the sprite from the MPEG compressed video with less computational burden. To achieve the above we have utilized the motion vector information for getting the foreground object mask and global motion information. The noise from the sparse motion vectors obtained from the predictive frames (P) are eliminated by considering the motion vectors extracted from the neighboring frames.

The rest of the paper is organized as follows. Section 2 explains the related works available in the literature. Section 3 describes the Overview of the proposed system. Section 4 talks about the pre-processing of motion vectors. Section 5 describes about coarse object segmentation. Section 6 discusses about the estimation of global motion parameters. Section 7 describes the sprite generation. Section 8 gives the experimental results. Finally, Section 9 concludes the paper.

2 Related Works

First work in literature which uses motion information for building mosaics was proposed by R. C. Jones *et al.* [2]. Here the camera motion parameters for a give shot is computed using the MPEG motion vectors, which is based on averaging the motion vectors from the macroblocks within each frame of MPEG video. The estimated parameters are used to integrate the frames with correction for zoom changes. Here the noisy motion vectors are directly used, which will lead to erroneous camera motion parameters. Moreover the effect of moving foreground objects are not considered in estimating the camera motion parameters. The work by Pilu [6] also used the motion vector information for determining the global camera motion by fitting the velocity field model. Here the motion vectors belonging to the low-textured macroblocks are not considered for the model fitting. Moreover in both the above mentioned works, the effects of moving foreground objects are not considered in estimating the camera motion parameters and suitable for the videos containing no moving foreground objects.

A. Smolic *et al.* [7] proposed a technique for estimation of long-term motion which describes the continuous and time-consistent motion over the whole sequence. A hierarchical long-term global motion estimation algorithm is used for generation of sprite. M. C. Lee *et al.* [4] developed a layered video object coding system with content based functionality using sprite and affine motion model. Both algorithms assume the availability of segmentation masks of foreground objects prior to sprite generation. Y. Lu *et al.* [5] proposed a sprite generation method with improved Global Motion Estimation (GME) and segmentation. A hybrid scheme with long-term and short-term motion estimation technique is used for determination of GME.

3 System Overview

The proposed system initially takes the compressed MPEG video shots and extracts motion vectors by partial decoding of the MPEG stream. The proposed system consists of the following four stages (i) Motion vector processing (ii) Coarse object segmentation (iii) Camera motion estimation (iv) Frame integration. Fig.1 shows the various parts of the proposed system.

Since the motion vectors obtained from the MPEG video are noisy, they can not be directly used for object segmentation. To increase the reliability of motion information, the motion vectors of few neighboring frames on either side of the current frame are also used. The details of this process are given in the next section. The segmentation stage takes the motion information obtained from the previous stage for segmenting the coherently moving video objects by simple K-means clustering. The number of objects in the video is determined by a proposed K-means clustering algorithm. The camera motion information is obtained from the motion information corresponding to the background object region. The estimated camera motion parameters are used for warping and blending the frames.

4 Processing the Motion Vectors

MPEG compressed video provides one motion vector (usually noisy) for each macro block of size 16×16 pixels. To remove noise, the motion vectors are subjected to the following steps (i) Motion Accumulation (ii) Determination of representative motion vectors as proposed in [1].

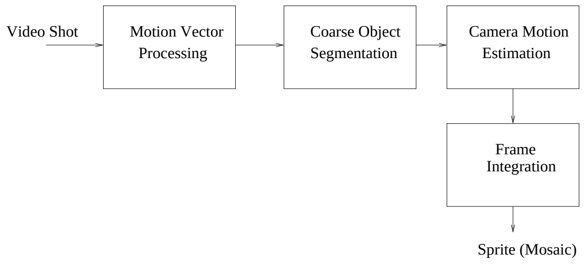


Figure 1: Overview of the proposed system

4.1 Motion Accumulation

Initially, the motion vectors are scaled appropriately to make them independent of frame type [3]. This is accomplished by dividing the motion vectors by the difference between the corresponding frame number and the reference frame number (in the display order). Then, the motion vectors are rounded to nearest integers. In the case of bidirectionally predicted macroblocks, reliable motion information is obtained either from the forward or backward motion vector depending upon which of the reference frames (I/P) is closer. If backward prediction is chosen, the sign of the motion vector is reversed after normalization.

The motion vector obtained from current macroblock of the current frame n is assigned to the center pixel (k, l) of that macroblock. Let $m_x^{kl}(n-c)$ and $m_y^{kl}(n-c)$ represent the motion vectors along the horizontal and vertical directions for the macroblock centered at (k, l) in frame $(n-c)$. Then, the new position for this macroblock in the current frame can be estimated as:

$$(\hat{k}, \hat{l}) = (k, l) + \sum_{f=n-1}^{n-c} \left(m_x^{kl}(f), m_y^{kl}(f)\right) \tag{1}$$

The motion vector $(m_x^{kl}(n-c), m_y^{kl}(n-c))$ in $(n-c)$th frame is assigned to the new position $(\hat{k}, \hat{l})$ with respect to the current frame. Fig. 2 explains the above process for the case of $c = 1$.

The motion accumulation is also done by tracking the frames in forward direction from the current frame. This is achieved by keeping few future frames in the buffer. In forward tracking the motion vectors are accumulated according to the following equation:

$$(\hat{k}, \hat{l}) = (k, l) - \sum_{f=n+1}^{n+c} \left(m_x^{kl}(f), m_y^{kl}(f)\right) \tag{2}$$

Here, the motion vector $(m_x^{kl}(n+c), m_y^{kl}(n+c))$ in $(n+c)$th frame is assigned to the new position $(\hat{k}, \hat{l})$ with respect to the current frame. Each frame approximately provides one additional motion vector per macroblock.

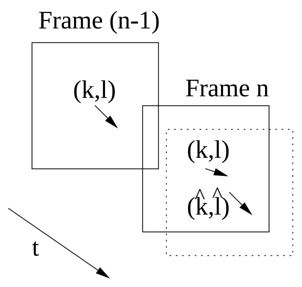


Figure 2: Relative motion of the macroblock of previous frame $n-1$ with respect to the corresponding macroblock of current frame n. The box drawn with dotted lines is the current position of the macroblock of previous frame.

4.2 Determination of representative motion vectors

After the motion accumulation process the representative motion vector for each macroblock is obtained by taking the median value of all the motion vectors falling within the corresponding macroblock region. The above process increases the reliability of the motion information by removing the noisy motion vectors present in the current frame. The representative motion vectors are given as input for the segmentation stage. Since we are not interested in extracting the exact shape of the object, this sparse motion information is sufficient to get the motion characteristics of the video object. Working with the sparse motion information reduces the computational burden involved in the segmentation process to a great extent.

5 Coarse Video Object Segmentation

The objective of this segmentation stage is to group together the coherently moving object blocks by applying K-means clustering technique. Given the number of objects N_o, the K-means clustering segments the entire frame into N_o coherently moving regions. Since the sprite generation, is done for frames whose background area is considerably more compared to the foreground objects, the cluster which occupies greater area is considered as background.

The number of motion models is to be determined before starting the segmentation process. The determination of the number of motion models is an important issue, because the final segmentation heavily depends upon the number of motion models. If the number of motion models is less, then the objects are merged, resulting in under segmentation. On the other hand if the number of motion models is more, then it results in splitting the objects which leads to over segmentation. All the motion models obtained from the motion accumulation phase are clustered using a K-means clustering algorithm by increasing the number of centers from one onwards and the mean square error (MSE) is observed. Since the clustering converges to a local minima, we use multiple restart for each number of clusters with randomly chosen cluster centers and pick the minimum value of MSE. The number of classes N_o, after which the decrease in MSE is less than a small threshold ζ, is chosen as the number of motion models. The typical value of ζ is chosen between 5 to 15 percent of the maximum error.

6 Camera Motion Estimation

Under the assumption that the camera is undergoing rotation and zoom but no translation, the change of image intensity between frames can be modeled by the following 6-parameter projective transformation [8].

$$x' = \frac{p_1 x + p_2 y + p_3}{p_5 x + p_6 y + 1} \tag{3}$$

$$y' = \frac{-p_2 x + p_1 y + p_4}{p_5 x + p_6 y + 1} \tag{4}$$

here, $p_1, \ldots p_6$ are the camera motion parameters and (x, y) and (x', y') are the image coordinates of the corresponding points in two neighboring frames with respect to the standard orthogonal set of axes with origin $(0, 0)$ at the image center. Suppose the camera undergoes small rotations (α, β, γ) about the X, Y and Z camera axes and the focal length changes from f to sf between two consecutive frames, then the parameters $p_1, \ldots p_6$ satisfy [8].

$$\begin{pmatrix} p_1 & p_1 & p_3 \\ -p_2 & p_1 & p_4 \\ p_5 & p_6 & 1 \end{pmatrix} = \begin{pmatrix} s & s\gamma & -sf\alpha \\ -s\gamma & s & sf\beta \\ \alpha/f & -\beta/f & 1 \end{pmatrix} \tag{5}$$

If we assume that the perspective distortion effects are negligible, then the parameters p_5 and p_6 can be set to zero. Now the rest of the parameters $p_1 \ldots p_4$ are estimated by the following weighted least squares technique.

$$\widehat{\boldsymbol{\theta}} = \mathbf{A}^{-1}\mathbf{B} \tag{6}$$

where,

$$\mathbf{A} =$$
$$\begin{pmatrix} \sum_i w_i(x_i^2 + y_i^2) & 0 & \sum_i w_i x_i & \sum_i w_i y_i \\ 0 & \sum_i w_i(x_i^2 + y_i^2) & \sum_i w_i y_i & -\sum_i w_i x_i \\ \sum_i w_i x_i & \sum_i w_i y_i & \sum w_i & 0 \\ \sum_i w_i y_i & -\sum_i w_i x_i & 0 & \sum w_i \end{pmatrix}$$

$$\mathbf{B} = \begin{pmatrix} \sum_i w_i(x_i' x_i + y_i' y_i) \\ \sum_i w_i(x_i' y_i - x_i y_i') \\ \sum_i w_i x_i' \\ \sum_i w_i y_i' \end{pmatrix}$$

and $\widehat{\boldsymbol{\theta}}^T = [p_1\ p_2\ p_3\ p_4]$. Here, p_1 indicates the inter-frame zoom factor ($p_1 > 1$,$p_1 = 1$ and $p_1 < 1$ correspond to zoom in, no zoom, zoom out respectively). The change in camera angle about the lens axis is given by the ratio p_2/p_1, camera pan and tilt rates are given by the ratios $-p_3/p_1$ and p_4/p_1 respectively. The weights w_i are determined with respect to the mode value of the motion vector angle. Since the camera motion is essentially reflected by the direction of the motion vector, the noisy motion informations are subjected to the following simple triangle M-estimator, which completely eliminates the highly corrupted motion vectors from camera motion estimation. The weight corresponding to the ith motion vector is given by

$$w_i = \hat{w}_i / \sum \hat{w}_i \tag{7}$$

where,

$$\hat{w}_i = \begin{cases} m\left(|\theta_i| - \theta_{mode}\right) + 1 & \text{if } |\theta_i| < \rho \\ 0 & \text{otherwise} \end{cases}$$

with slope $m = -\frac{1}{\rho}$; where ρ gives the range of angle on either side of the mode angle to be considered for global motion estimation.

In most of the cases the camera motion is captured by the background object which usually occupies more area than any other object. Hence, only the motion vectors corresponding to the biggest object in the frame is considered for camera motion estimation and the motion vectors corresponding to the other objects are ignored.

$$\mathbf{w}'_k = \begin{pmatrix} x'_k \\ y'_k \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \left[\begin{pmatrix} i_k \\ j_k \end{pmatrix} - \begin{pmatrix} i_o \\ j_o \end{pmatrix} \right] \tag{8}$$

Here, $\mathbf{w}'_k$ represents the new location of the kth accumulated motion vector (m_x^k, m_y^k) with respect to the image-centered Cartesian axis, whose original location in the current frame is (i_k, j_k) and (i_o, j_o) is the coordinates of the center of the image. The corresponding locations of the points (x_k, y_k) in the previous frame for the current frame point (x'_k, y'_k) is given by

$$\mathbf{w}_k = \begin{pmatrix} x_k \\ y_k \end{pmatrix} = \begin{pmatrix} x'_k \\ y'_k \end{pmatrix} + \begin{pmatrix} m_x^k \\ -m_y^k \end{pmatrix} \tag{9}$$

7 Sprite Generation

For sprite generation we only use the motion vectors decoded from the P frames of the given video sequence. The system aligns frames directly from the camera motion estimates by avoiding complicated registration of frames. The outline of the algorithm is described as follows:

- In this work the first P frame is used as the reference frame for sprite.
- The background object mask $\mathbf{M}_1$ of the first $P(1)$ frame is obtained as explained in Section 5 using the processed motion vectors.
- Possible foreground object parts near the background region boundaries are removed by eroding the background mask to a depth of 4-6 pixels.
- only the pixels that belong to the background object is used for constructing the sprite $\mathbf{S}_1$, leaving the foreground/unreliable regions to be filled by the future P frames.
- The following P frames are also subjected to the coarse object segmentation. Before integrating the background region of nth P frame ($P(n)$) with the sprite, the frame should be warped to the existing sprite. The warping parameters are updated as follows:

$$\begin{aligned} p_1(n) &= \prod_i^n p_1(i) \\ p_2(n) &= \prod_i^n p_2(i) \\ p_3(n) &= \sum_i^n \frac{p_3(i)}{p_1(i)} \\ p_4(n) &= \sum_i^n \frac{p_4(i)}{p_1(i)} \end{aligned} \tag{10}$$

the frame $P(n)$ is warped according to:

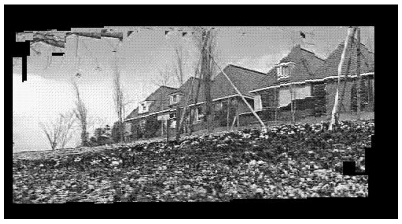
$$\begin{pmatrix} x_k \\ y_k \end{pmatrix} = \begin{pmatrix} p_1 & p_2 \\ -p_2 & p_1 \end{pmatrix}^{-1} \left[\begin{pmatrix} x'_k \\ y'_k \end{pmatrix} - \begin{pmatrix} p_3 \\ p_4 \end{pmatrix} \right] \tag{11}$$

Since the motion vectors give the correspondence between the current frame and the reference frame, the translational motion parameters p_1, p_2 are corrected with respect to the recent sprite data. Where as the sprite is generated from many past P frames. So to get the proper matching between the sprite and the current P frame, the current frame background object is warped towards the sprite and matched against the generated sprite. The matching uses the MAD criterion. The correction for p_3 and p_4 are given by,

$$[c_1\ c_2] = \arg \min_{(c_1,c_2)} MAD(c_1, c_2) \tag{12}$$

where,

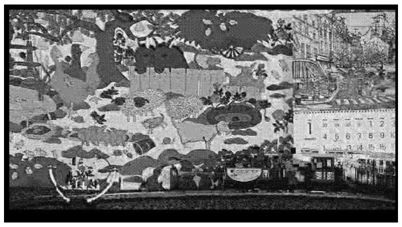
$$MAD(c_1, c_2) = \sum_{(n_1,n_2)\in\mathcal{B}} \left| P^n(n1, n2) - \mathbf{S}^{n-1}(n1 + c1, n2 + c2) \right| \tag{13}$$



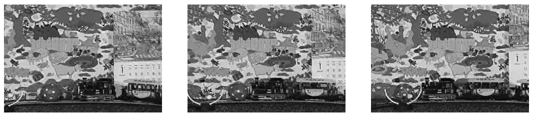
(a)

(b)

Figure 3: (a) Sprite generated for the flower garden sequence (b) 5th, 45th and 100th frames of flower garden sequence



(a)



(b)

Figure 4: (a) Sprite generated for the mobile sequence (b) 25th, 75th and 125th frames of mobile sequence

with $-M \leq c_1, c_2 \leq M$ (typical value of M is around 2) and the region $\mathcal{B}$ contains the background region common to the current P^n frame and the sprite $\mathbf{S}^{n-1}$, excluding the foreground object, unreliable border regions and the region uncovered by the current P^n frame. Now the corrected parameters are given by $p_3' = p_3 - c_2$ and $p_4' = p_4 + c_1$. Only the regions uncovered by the background object of the latest frame is integrated with the existing sprite by warping with the new parameters, to get the new sprite $\mathbf{S}_n$.

8 Results and Discussion

To demonstrate the performance of the mosaicing technique we ran the algorithm on several video shots. The results are shown in Figs. 3- 6. For all sequences only the P frames are used for estimating the global motion and generating the sprite. The GOP structure for bike sequence is *IBBPBBIBBP*... and *IBBPBBPBBPBBPBBI*... for other sequences. The dark regions within the sprite are due to the background regions occluded by the moving foreground objects which are never uncovered. In all these sequences there are at least one independently moving foreground object which occludes the background object. The results show the feasibility of constructing the sprite image of the MPEG shot with very less computational effort by using the motion information which is readily available in MPEG compressed stream.

9 Conclusions

In this paper we have presented a faster technique for background sprite generation from the MPEG video. The foreground objects are segmented out in compressed domain using the motion vectors. Only the motion information available in the P frames are used for segmentation and sprite generation. The global motion parameters are obtained from the motion information corresponding to the background object and avoids the computationally intensive motion estimation step. Moreover, since all the computations are based on the sparse motion vectors, the complexity of the system is reduced very much. This method of sprite generation can be extended to browsing and indexing large video databases.

References

[1] R. V. Babu and K. R. Ramakrishnan. Content-based video retrieval using motion descriptors extracted from compressed domain. In *IEEE ISCAS*, May 2002.

[2] R. C. Jones, D. DeMenthon, and D. S. Doermann. Building mosaics from video using MPEG motion vectors. In *ACM Multimedia (2)*, pages 29–32, 1999.

[3] V. Kobla, D. S. Doermann, and K.-I. Lin. Archiving, indexing and retrieval of video in compressed domain. In *SPIE Conference on Multimedia Storage and Archiving Systems*, volume 2916, pages 78–89, 1996.

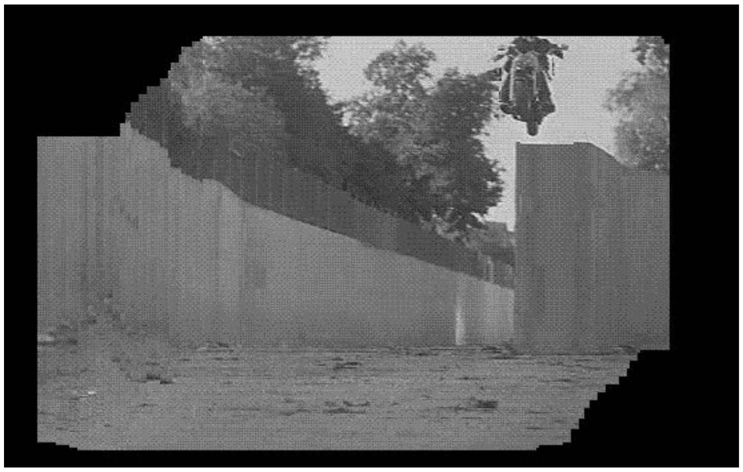
[4] M. C. Lee, W. Chen, C. B. lin, C. Gu, T. Markoc, S. I. Zabinsky, and R. Szeliski. A layered video object coding system using sprite and affine motion model. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(1):130–145, Feb. 1997.

[5] Y. Lu, W. Gao, and F. Wu. Sprite generation for frame-based video coding. In *IEEE ICIP*, pages 473–476, 2001.

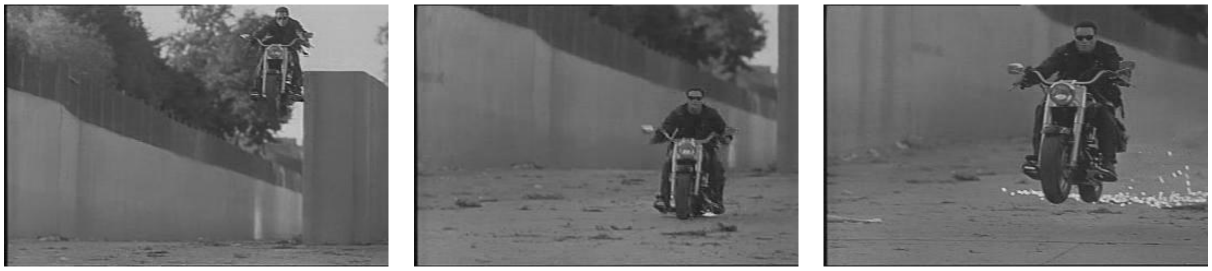
[6] M. Pilu. On using raw MPEG motion vectors to determine global camera motion. Technical Report HPL-97-102, HP Laboratories, Sept. 1997.

[7] A. Smolic, T. Sikora, and J. R. Ohm. Long-term global motion estimation and its application for sprite coding, content description and segmentation. *IEEE Trans. on Circuits and Systems for Video Technology*, 9(8):1227–1242, Dec. 1999.

[8] Y. P. Tan, D. D. Saur, S. R. Kulkarni, and P. J. Ramadge. Rapid estimation of camera motion from compresses video with applications to video annotation. *IEEE Trans. on Circuits and Systems for Video Technology*, 10(1):133–146, Feb. 2000.

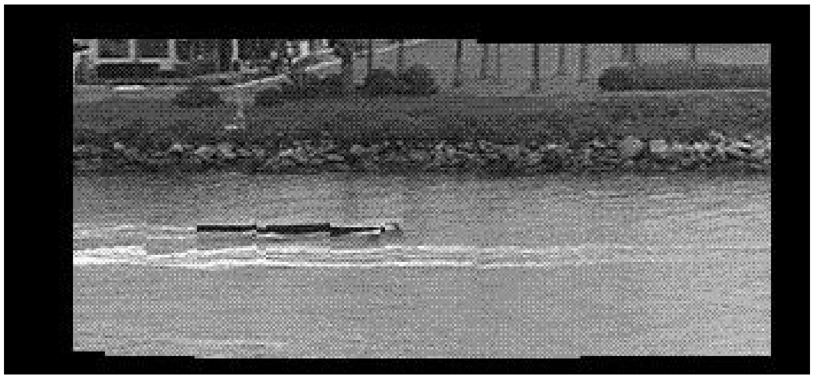


(a)

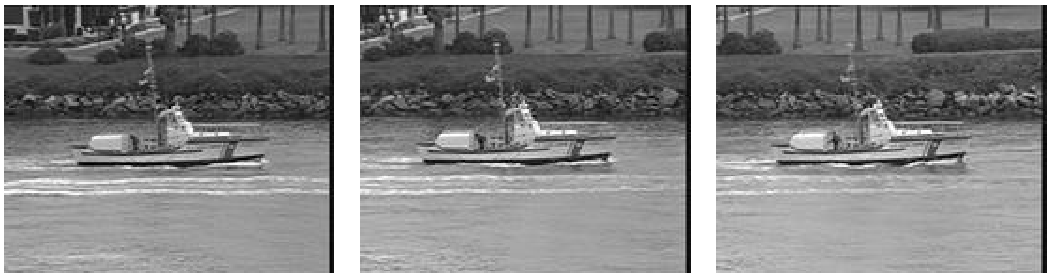


(b)

Figure 5: (a) Sprite generated for the bike sequence (b) 50th, 90th and 130th frames of the bike sequence



(a)



(b)

Figure 6: (a) Sprite generated for the boat sequence (b) 45th, 90th and 140th frames of the boat sequence